A Fast Tree Algorithm for Electric Field Calculation in Electrical Discharge Simulations

Chijie Zhuang[1], Yong Zhang[2], Xin Zhou[1], Rong Zeng[1], Jinliang He[1], and Lei Liu[3]
[1]Department of Electrical Engineering, Tsinghua University, Beijing 100084, China.
[2]Courant Institute of Mathematical Sciences, New York University, New York, NY 10011, USA.
[3]China Southern Grid Electric Power Research Institute, Guangzhou, 510663, China.

The simulation of electrical discharges has been attracting a great deal of attention. In such simulations, the electric field computation dominates the computational time. In this paper, we propose a fast tree algorithm that helps to reduce the time complexity from $O(N^2)$ (from using direct summation) to $O(N \log N)$. The implementation details are discussed and the time complexity is analyzed. A rigorous error estimation shows the relative error of the tree algorithm decays exponentially with the number of truncation terms and can be controlled adaptively. Numerical examples are presented to validate the accuracy and efficiency of the algorithm.

***Index Terms*—tree algorithm, electric field, electrical discharge, disc model, error estimation.**

I. Introduction

THERE are various types of electrical discharges in nature, e.g., lightning strikes [1], corona discharges around electrodes in non-uniform electric fields [2]. Because of the relevance of electrical discharge to everyday life and its growing application in industry, the numerical simulation of electrical discharges has been increasingly attracting attention.

The most widely adopted model for electrical discharge simulations is the fluid model [3], [4]. This model consists of the Poisson equation, which describes the electric field that drives the electrical discharge, and the convection-diffusion equations with source terms, which describe the charge-carrier transport.

Because of its high computational load, the simulation of electrical discharge under atmospheric pressure is, at present, mainly limited to short gap discharges of a few centimeters in length [4]. Thus, many simplified models have been proposed in the hope of simulating longer discharges, e.g., 100 cm in length. Among these models, the most promising one is the so-called 1.5-dimensional model [5].

In the 1.5-dimensional model, the charges are assumed to be distributed among discs of the same radius. On each disc the charge density is uniform, and the charges only move along the y-axis (see Fig. 1). The charge transport is described using a one-dimensional model, while the electric field is considered to be two-dimensional. Using this so-called disc method, the electric field can be derived analytically. Assume there is a disc of net charge density $\sigma(x)$, radius r_d, thickness $\mathrm{d}x$ (see Fig. 1). The electric field it generates at a point, y, along the y-axis is given by:

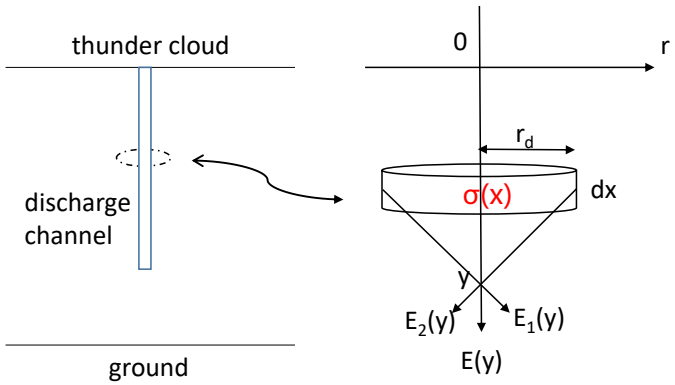

Fig. 1. Diagram of a 1.5-dimensional model

$$\mathrm{d}E(y) = \begin{cases} \frac{\sigma(x)}{2\varepsilon_0}\left(\frac{x-y}{\sqrt{(y-x)^2+r_d^2}}+1\right)\mathrm{d}x, & x-y<0; \\ \frac{\sigma(x)}{2\varepsilon_0}\left(\frac{x-y}{\sqrt{(y-x)^2+r_d^2}}-1\right)\mathrm{d}x, & x-y\geq 0. \end{cases} \quad (1)$$

To consider the influence of the electrodes on the electric field, all image charges, e.g., which are above the cloud and below the ground in Fig. 1, should be taken into account. However, only image charges whose distances to the electrodes are less than the discharge-gap length L are considered because image charges that are far way contribute little to the electric field. Integrating over the whole domain, we get

$$E(y) = \frac{1}{2\varepsilon_0}\left[\int_{-L}^{y}\sigma(x)\left(\frac{x-y}{\sqrt{(x-y)^2+r_d^2}}+1\right)\mathrm{d}x + \int_{y}^{L}\sigma(x)\left(\frac{x-y}{\sqrt{(x-y)^2+r_d^2}}-1\right)\mathrm{d}x\right]. \quad (2)$$

Assuming there are N source charges and N target points, the computation of Eq. (2) has a time complexity of $O(N^2)$. As a result, the electric field evaluation may occupy around 90% of the CPU time in a simulation [6], and fast algorithms with better complexity are highly imperative.

Manuscript received xxx xxx, xxxx; revised xxx xxx, xxx; accepted xxxx xx, xxxx. Date of publication xxx xxx, xxxx; date of current version xxx xxx, 2017. Corresponding authors: Yong Zhang, Rong Zeng and Chijie Zhuang (e-mail: sunny5zhang@gmail.com, zengrong@tsinghua.edu.cn, chijie@tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier xxxxxxx.xxxxx

In fact, there has been several works on such fast evaluation of potential and field, e.g., the Barnes–Hut fast tree algorithm [7], and the famous fast multipole method [8], [9] for N-body simulation. In this paper, we propose a tree algorithm for the specific kernel arising from electrical discharge simulations, employing the same ideas of far-field, near-field evaluation, which dramatically helps to accelerate the field evaluation with a highly controllable accuracy [7], [10].

II. The Tree Algorithm

By integrating Eq. (2) using sufficient high-order Gaussian quadrature, and setting $q_j := \frac{\omega_j \sigma_j}{2\varepsilon_0}\Delta x$ where ω_j is the associated weight of the Gaussian quadrature and Δx is the length of the associated interval, Eq. (2) can be reduced to

$$\begin{aligned} E(y) &= \big(\sum_{j=0}^{m} q_j - \sum_{j=m+1}^{n} q_j\big) + \sum_{j=0}^{n} \frac{q_j(x_j - y)}{\sqrt{(x_j-y)^2 + r_d^2}} \\ &:= e_m + \sum_{j=0}^{n} \frac{q_j(x_j - y)}{\sqrt{(x_j-y)^2 + r_d^2}}. \end{aligned} \tag{3}$$

where $e_m = \sum_{j=0}^{m} q_j - \sum_{j=m+1}^{n} q_j$. The term e_m can be calculated recursively, i.e.,

$$e_{m+1} = e_m + 2q_{m+1}.$$

Thus e_0 is computed first, followed by the successive calculation of e_1, e_2..., e_n. This work has a linear time complexity. Below we will omit the term e_m for brevity, but the principle of the tree algorithm remains unchanged.

As shown in Fig. 2, the total electric field, E, is split into two parts, i.e., the far-field E_f and the near-field E_n such that $E = E_f + E_n$. The fundamental idea of the tree algorithm is that the far-field interaction, which is from the charges far away from the target point, is approximated as if they are a group, while the near-field from the neighboring charges is evaluated directly.

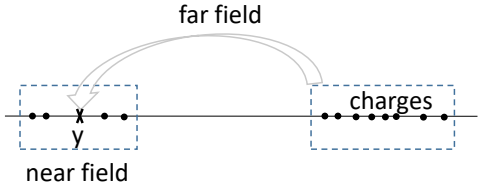

Fig. 2. Diagram of the near-field and far-field interactions

Assume a cluster of charges $\{q_j\}_{j=0}^n$ located at $\{x_j\}_{j=0}^n$ are gathered around x_c, and $|y - x_c| \gg 0$, $|y - r_d| \gg 0$. To calculate the far-field $E_f(y)$, a crude approximation is

$$E_f(y) = \sum_{j=0}^{n} q_j \Phi(x_j, y) \approx \big(\sum_{j=0}^{n} q_j\big)\Phi(x_c, y), \tag{4}$$

with $\Phi(x,y) := \frac{x-y}{\sqrt{(x-y)^2+r_d^2}}$. However, using Taylor expansion, we have

$$\begin{aligned} \Phi(x,y) &= \sum_{k=0}^{\infty} \frac{1}{k!}\Phi^{(k)}(x_c, y)(x - x_c)^k \\ &= \sum_{k=0}^{p} \frac{1}{k!}\Phi^{(k)}(x_c, y)(x - x_c)^k + R_p(x), \end{aligned} \tag{5}$$

where $\Phi^{(k)} = \frac{\partial^k \Phi}{\partial x^k}$; $p \in \mathbb{N}$; the residual R_p is given by $R_p = \sum_{k=p+1}^{\infty} \frac{1}{k!}\Phi^{(k)}(x_c, y)(x - x_c)^k$. Therefore, we have

$$\begin{aligned} E_f(y) &= \sum_{j=0}^{n} q_j \left(\sum_{k=0}^{\infty} \frac{1}{k!}\Phi^{(k)}(x_c, y)(x_j - x_c)^k\right) \\ &\approx \sum_{k=0}^{p} \Phi^{(k)}(x_c, y) \left(\sum_{j=0}^{n} q_j \frac{(x_j - x_c)^k}{k!}\right). \end{aligned} \tag{6}$$

When $p = 0$, Eq. (6) reduces to the crude approximation Eq. (4). To approximately calculate $E_f(y)$, one only needs to calculate the moments $\left(\sum_{j=0}^{n} q_j \frac{(x_j - x_c)^k}{k!}\right)$ and $\Phi^{(k)}(x_c, y)$, for $k = 0, ...p$.

We now derive a recurrence formula to calculate $\Phi^{(k)}(x, y)$. It is straightforward that

$$\Phi^{(0)}(x,y) = \frac{x-y}{\sqrt{(x-y)^2+r_d^2}}, \tag{7}$$

$$\Phi^{(1)}(x,y) = \frac{r_d^2}{\left(\sqrt{(x-y)^2+r_d^2}\right)^3}, \tag{8}$$

which implies that

$$r_d^2 \Phi^{(0)}(x,y) = \Phi^{(1)}(x,y)\big[(x-y)^3 + r_d^2(x-y)\big]. \tag{9}$$

Differentiating Eq. (9) for k-1 using the general Leibniz rule, after some algebraic simplifications, we get

$$\begin{aligned} &(x-y)[(x-y)^2 + r_d^2]\Phi^{(k)}(x,y) = \\ &\big[r_d^2 - (k-1)(3(x-y)^2 + r_d^2)\big]\Phi^{(k-1)}(x,y) \\ &-3(k-1)(k-2)(x-y)\Phi^{(k-2)}(x,y) \\ &-(k-1)(k-2)(k-3)\Phi^{(k-3)}(x,y). \end{aligned} \tag{10}$$

Therefore, by using Eqs. (7) to (10), for any given y, $\Phi^{(k)}(x_c, y)$ may be calculated recursively for $k = 2, 3, ...p$.

III. Error Estimation

Now we present a rigorous error estimation for Eq. (6). Without loss of generality, we only consider the case $x_c = 0$. Other cases reduce to the $x_c = 0$ case after a simple shift, i.e. let $x := x - x_c$.

Define a complex function $f(z) := \frac{z-y}{\sqrt{(z-y)^2+r_d^2}}$ with $z \in \mathbb{C}$, which is analytic for $|z| < \sqrt{y^2 + r_d^2}$. By Cauchy's integral formula, for any z satisfying $|z| := r \le R := |y|$,

$$f(z) = \frac{1}{2\pi i}\oint_\Gamma \frac{f(\xi)}{\xi - z}\mathrm{d}\xi, \qquad f^{(k)}(0) = \frac{k!}{2\pi i}\oint_\Gamma \frac{f(\xi)}{\xi^{k+1}}\mathrm{d}\xi, \tag{11}$$

where $i = \sqrt{-1}$, $\Gamma := \{w \in \mathbb{C} || w| = R\}$ is a contour containing the point z. We have

$$\begin{aligned} f(z) &= \frac{1}{2\pi i}\oint_\Gamma \frac{f(\xi)}{\xi}\frac{1}{1 - \frac{z}{\xi}}\mathrm{d}\xi \\ &= \frac{1}{2\pi i}\oint_\Gamma \frac{f(\xi)}{\xi}\big(\sum_{k=0}^{p}(\frac{z}{\xi})^k + \frac{(\frac{z}{\xi})^{p+1}}{1 - \frac{z}{\xi}}\big)\mathrm{d}\xi \\ &= \frac{1}{2\pi i}\big(\sum_{k=0}^{p} z^k \oint_\Gamma \frac{f(\xi)}{\xi^{k+1}}\mathrm{d}\xi + \oint_\Gamma \frac{f(\xi)}{\xi}\frac{(\frac{z}{\xi})^{p+1}}{1 - \frac{z}{\xi}}\mathrm{d}\xi\big) \\ &= \sum_{k=0}^{p} \frac{f^{(k)}(0)}{k!}z^k + \frac{1}{2\pi i}\oint_\Gamma \frac{f(\xi)}{\xi}\frac{(\frac{z}{\xi})^{p+1}}{1 - \frac{z}{\xi}}\mathrm{d}\xi. \end{aligned} \tag{12}$$

Comparing $f(z)$ and $\Phi(x)$, we find that $\Phi(x) = f(z)|_{z=x}$, so

$$|R_p| = \left| \frac{1}{2\pi i} \oint_\Gamma \frac{f(\xi)}{\xi} \frac{(\frac{z}{\xi})^{p+1}}{1-\frac{z}{\xi}} \mathrm{d}\xi \right|_{z=x} \leq \max \left\{ \left| \frac{1}{2\pi i} \oint_\Gamma \frac{f(\xi)}{\xi} \frac{(\frac{z}{\xi})^{p+1}}{1-\frac{z}{\xi}} \mathrm{d}\xi \right| \right\}. \quad (13)$$

Using the fact $|f(\xi)|$ is bounded for $\xi \in \Gamma$, i.e. $|f(\xi)| \leq M$, we get

$$\begin{aligned} |R_p| &\leq \frac{1}{2\pi} \oint_\Gamma \max \left(\left| \frac{(\frac{z}{\xi})^{p+1}}{1-\frac{z}{\xi}} \right| \left| \frac{f(\xi)}{\xi} \right| \right) \mathrm{d}\xi \\ &\leq \max \left(\left| \frac{(\frac{z}{\xi})^{p+1}}{1-\frac{z}{\xi}} \right| \right) \max |f(\xi)| \\ &\leq M \frac{R}{R-r} (\frac{r}{R})^{p+1}. \end{aligned} \quad (14)$$

Eq. (14) shows Eq. (6) converges as p increases if $r < R$, which is easy to be satisfied; to be more precise, the error decays exponentially with respect to p. As an example, $|R_p|$ is sufficiently small when $p = 15$ or 20 if $\frac{r}{R} \leq \frac{1}{3}$.

IV. Implementation and Efficiency Analysis

Equation (6) is used to approximate the far field when the target and sources points are well separated. Now we illustrate in Fig. 3 how to determine whether the target and source charges are well separated. In Fig. 3, three intervals, all with a diameter of $2r$, are shown. The target point, y_0, lies in cell 1, and its distances to centres of cell 2 and 3 are R_1 and R_2 respectively. We say that cell 1 and cell 2 are direct neighbors if $\frac{r}{R_1} > \frac{1}{3}$; while cell 1 and cell 3 are well separated if and only if $\frac{r}{R_2} \leq \frac{1}{3}$.

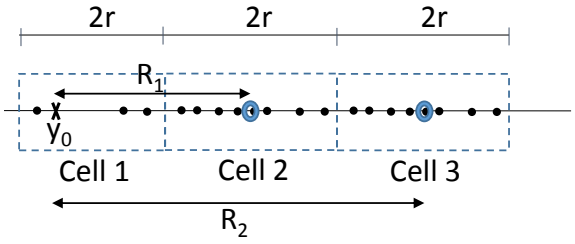

Fig. 3. Diagram showing direct and well-separated neighbors

We now build a binary tree to successively approximate the far field, an example of which can be seen in Fig. 4. For simplicity, all the sources are assumed to be in [0,1] and the target is assumed to be in $[\frac{1}{8}, \frac{3}{16}]$. The target point and all the charges are direct neighbors at the first two levels. In level 2, the target and $(\frac{1}{2}, \frac{3}{4}]$, $(\frac{3}{4}, 1]$ are well separated while all others remain direct neighbors. The intervals are further subdivided, which results in $(\frac{3}{8}, \frac{1}{2}]$ becoming the well separated neighbor.

This process is repeated until the bottom level is reached. There are finally at most two direct neighbors of the interval containing the target point, while the very interval and all other intervals are well separated. The near field from the interval containing the target point and the direct neighbors, is

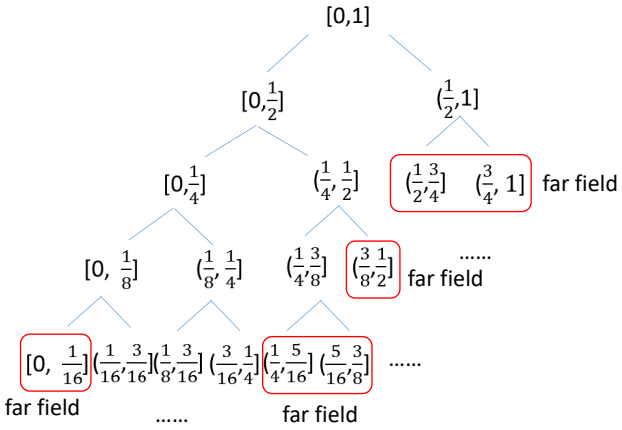

Fig. 4. Diagram of the binary tree structure

evaluated directly; while the far field from other well separated intervals at different levels are approximated by Eq. (6).

Assuming c is the number of source charges in an interval at the bottom level of the tree, then c is $O(1)$ and $2^m \approx N/c$, which implies m is $O(\log N)$.

Now we are ready to estimate the computational cost for a single target point, ignoring the cost of setup. The work for the far field part involves the evaluation of at most three far field expansions of p terms at each level from 2 to m. Therefore, the flop count arising from the evaluation of the far field expansions is $O(mp)$. The near field evaluation, which is done at the bottom level, requires at most three intervals. Since each bottom level interval contains only $O(1)$ sources, the flop count of the direct calculation is $O(1)$. Hence the cost in flops of an evaluation for one single target point is $O(mp) \approx O(p \log N)$, which is typically much faster than the $O(N)$ flop count associated with direct summation.

At each level, the setup cost, which is mainly the formation of the moments in Eq. (6), is at most $O(pN)$, so the total setup cost is at most $O(mN) \approx O(N \log N)$.

Overall, the computational cost of the algorithm is $O(N \log N)$ for N targets.

V. Validation and Efficiency

A. Variation of the error with the number of truncation terms

First, only the far-field was calculated to validate Eq. (6). We randomly generated 10000 charges in $[-0.5, 0.5]$, which is around $x = 0$, and set $r_d = 0.1$, The electric field at $y = 1$ $(r/R = 0.5)$ was then calculated using different numbers of truncation terms, denoted by p. Results in Tab. I show the relative error decays exponentially with p, which coincides with the error estimation in Eq. (14). When p increases by five, the error decays by a factor of about 50-100.

TABLE I
Accuracy with different numbers of truncation terms (p)

p	5	10	15	20
relative error	9.43e-5	1.17e-6	6.40e-8	6.48e-10

B. Impact of the number of tree levels on efficiency

Next, the impact of the number of tree levels on the CPU calculation time was tested. The algorithm was implemented in C++, and the experiments here and below were performed on a PC with an i7-6500U CPU and 8 GB RAM. With more levels, the effect of more sources are calculated by Eq. (6), which may accelerate the computation; however, more tree levels are traversed, which may slow down the computation. In our experiment, 2×10^5 charge sources, each with a random amount of charge, were uniformly randomly distributed in $[0,1]$, the target and source positions were the same. The field generated by the neighboring charges were calculated directly and others by Eq. (6) with $p=10$.

It is shown in Tab. II that the depth of the tree or, in other words, the number of particles in the bottom-level interval, greatly influences the computational efficiency. Our test shows that the best number of particles is about 48 for $p=10$.

TABLE II
COMPUTATIONAL TIME WITH DIFFERENT DEPTHS($p=10$, $N=2\times10^5$)

levels	9	10	11	12	13	14	15	16
# particles	781	390	195	97	48	24	12	6
time (ms)	2418	1450	1014	827	765	780	795	842

particles means the estimated number of the particles in one interval at the bottom level of the tree.

C. Efficiency with different number of particles and targets

After optimizing the number of particles in the bottom-level interval, both the near-field and far-field were evaluated in order to test the efficiency of the tree algorithm. Different numbers of charge sources, each with a random amount of charge, were randomly placed in $[0,1]$, the evaluation locations were the same as the source positions. The tree levels were determined such that the finest interval contained about 40 particles. The other configuration were the same as in the above experiment and the experiments were repeated multiple times. The results in Tab. III show that the time complexity of the algorithm is roughly $O(N\log N)$, which is much faster than direct summation even when N is small.

TABLE III
TIME-COST COMPARISONS FOR DIFFERENT NUMBERS OF CHARGES AND TARGETS

particles	time by tree algorithm (ms)			time by direct summation (ms)		
	max	min	average	max	min	average
1e4	29	24	27.4	577	453	505.5
5e4	170	156	160.4	11840	11060	11372.8
10e4	385	353	362.8	48518	44058	45929.4
15e4	587	571	577.4	104083	100843	101882.8
20e4	858	674	772.8	188105	180105	182944.8

In addition, two types of errors, maximal (left) and average (right), were measured:

$$\max_i \left|\frac{E_i^{\text{tree}}-E_i^{\text{dir}}}{E_i^{\text{dir}}}\right|, \qquad \frac{\sum_i(|E_i^{\text{tree}}-E_i^{\text{dir}}|)}{\sum_i|E_i^{\text{dir}}|}.$$

Table IV shows the maximal and average errors are roughly of the same order for different numbers of particles and targets, and are very small, which infers that the algorithm is reliable.

TABLE IV
ACCURACY FOR DIFFERENT NUMBERS OF CHARGES AND TARGETS

# of particles	1e4	5e4	10e4	15e4	20e4
maximal error	2.89e-10	4.41e-10	2.44e-9	1.61e-9	2.75e-9
average error	6.16e-14	4.96e-14	4.90e-14	4.75e-14	5.25e-14

# of particles means the total number of particles.

VI. CONCLUSION

We present in this paper a fast tree algorithm of $O(N\log N)$ complexity to calculate the electric field arising from 1.5 dimensional electrical discharge simulations.

The tree algorithm is derived based on Taylor expansion. A recurrence formula following the general Leibniz rule is provided to calculate the expansion coefficients efficiently.

Error estimation shows that error decays exponentially as the number of truncation terms increases, and detailed analysis confirmed the $O(N\log N)$ time complexity with tunable accuracy, which represents a dramatic improvement over direct summation method. Numerical experiments were given to validate the efficiency and accuracy.

Developing fast algorithms of linear time complexity following the ideas of fast multipole method will be our further direction, especially in higher space dimensions.

ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China under grant 51577098 and grant 51325703, Open Fund of National Engineering Laboratory for Ultra High Voltage Engineering Technology (Kunming, Guangzhou), the Schrödinger Fellowship J3784-N32. Prof. Jingfang Huang at University of North Carolina, Chapel Hill, is greatly acknowledged for continuous help during the years.

REFERENCES

[1] Rong Zeng, Chijie Zhuang, Xuan Zhou, She Chen, Zezhong Wang, Zhanqing Yu, Jinliang He. Survey of recent progress on lightning and lightning protection research. High Voltage. 2016, 1(1): 2-10.

[2] Yuesheng Zheng, Bo Zhang, Jinliang He. Onset conditions for positive direct current corona discharge under the action of photoionization. Physics of Plasmas. 2011, 18: 123503.

[3] Chijie Zhuang, Rong Zeng, Bo Zhang, Jinliang He. 2-D discontinuous Galerkin method for streamer discharge simulations in Nitrogen. IEEE Transactions on Magnetics. 2013, 49(5): 1929-1932.

[4] A. Luque, U. Ebert. Density models for streamer discharges: Beyond cylindrical symmetry and homogeneous media. Journal of Computational Physics. 2012, 231, 904-918.

[5] Chijie Zhuang, Rong Zeng. A local discontinuous Galerkin method for 1.5-dimensional streamer discharge simulations. Applied Mathematics and Computation. 2013, 219: 9925-9934.

[6] S. Pancheshnyi, P. Ségur, J. Capeillère, A. Bourdon. Numerical simulation of filamentary discharges with parallel adaptive mesh refinement. Journal of Computational Physics, 2008, 227: 6574-6590.

[7] Josh Barnes, Piet Hut. A hierarchical O(NlogN) force-calculation algorithm. Nature. 1986, 324: 446-449.

[8] L. Greengard, V. Rokhlin. A fast algorithm for particle simulations, Journal of Computational Physics. 1987, 73: 325-348.

[9] Rick Beatson, Leslie Greengard. A short course on fast multipole methods. Wavelets, Multilevel Methods and Elliptic PDEs. 1997, 150: 1-37.

[10] Keith Lindsay, Robert Krasny. A particle method and adaptive treecode for vortex sheet motion in three-dimensional flow. Journal of Computational Physics. 2001, 172: 879-907.